Antonio Stanciu, Marti Girón, Roger Guiñón

Forex Forecasting with Sentiment Analysis: EUR/USD Case Study

Introduction

Forecasting the short-term direction of the foreign exchange (Forex) market is extremely difficult due to its enormous liquidity and volatility. Because of the numerous influencing factors and noise, it is particularly challenging to predict the daily movement (increase, fall, or slight shift) of a major currency pair like EUR/USD. Statement of the Problem: In order to better classify daily EUR/USD swings into BUY, HOLD, or SELL signals, this study investigates if adding news sentiment to a price-based predictive model can be beneficial. According to the premise, combining sentiment from news with conventional market data will improve predictive value over relying solely on price data. In order to test this, we create a repeatable pipeline that combines sentiment from global news with historical price data, and we assess two modeling strategies (a deep learning model and a tree-based model) on a three-class classification job.

Goals: (1) collect and process a suitable dataset of market data (EUR/USD prices) and news sentiment (from the GDELT database) over a period of several years; (2) use a threshold-based rule from the literature to define a labeling strategy for daily price direction (up, down, or no significant change); (3) design and implement two predictive models, an XGBoost ensemble and an LSTM neural network, and justify their configurations; and (4) assess whether adding sentiment features enhances prediction performance in comparison to a price-only baseline, evaluating outcomes with relevant metrics and visualizations. Documenting the project's development, including early model iterations, problems (such class imbalance and data alignment), and later enhancements, is another goal. The final objective is to address the limitations and potential directions for future research while offering insight into the effectiveness of news sentiment for Forex forecasting.

Data and Preprocessing

Pricing Data: We employ daily historical pricing data for the EUR/USD currency pair, spanning from 2013 to early 2025 (roughly 12 years). Daily data comprises open, high, low, and close (OHLC) prices, utilized to compute the daily log-return of the closing price. Employing log returns (the natural logarithm of close_t / close_{t-1}) is a conventional method to standardize price fluctuations and address non-stationarity in financial time series. The pricing data was sourced from a provider, such as Alpha Vantage, and encompasses diverse market situations, including significant economic events, so assuring a comprehensive dataset for modeling.

News Sentiment Data: We utilize the Global Database of Events, Language, and Tone (GDELT) to measure market sentiment, as it is an extensive news database that documents events and their corresponding tone (sentiment) from global news sources. We extracted around 22,000 news headlines and articles from GDELT between 2013 and 2025 that pertain to the Forex market, primarily concentrating on USD, EUR, or macroeconomic news impacting these currencies. The text of each news item was analyzed using FinBERT, a BERT-based language model pre-trained on

financial corpora for sentiment detection. FinBERT is tailored for financial language and has demonstrated superior performance compared to general sentiment models in finance-specific sentiment tasks. We utilized FinBERT on each news article to derive a sentiment score, a continuous polarity metric where positive values signify bullish sentiment, negative values denote bearish sentiment, and values around zero reflect neutral sentiment. Sentiment ratings from many news articles on the same day were subsequently consolidated into a singular daily sentiment metric (e.g., through averaging) to correspond with daily price data. This produces a time series of sentiment values corresponding to the price data.

The price and sentiment datasets were integrated based on the date field to form a consolidated daily dataframe. A crucial preprocessing step involved managing dates where one source contains data while the other does not. For example, on weekends or holidays, the market is closed (resulting in the absence of price data), while news mood may still be documented, and conversely. Our technique involved synchronizing all elements with the trading days calendar: sentiment values from non-trading days are propagated to the subsequent trading day (forward-fill imputation) based on the premise that the sentiment impact from a weekend extends into Monday in the absence of new signals. This guarantees that each trading day possesses an associated emotion value. Furthermore, any absent mood on a trading day (in the absence of captured news) is classified as neutral (e.g., zero or carried forward from the previous day). Subsequent to merging, we standardized and normalized characteristics as necessary. Specifically, continuous variables such as mood and technical indicators were normalized (z-scored) to prevent any single characteristic from dominating due to discrepancies in size. This is particularly significant for the neural network model.

Feature Engineering: In addition to the fundamental daily log return and unprocessed emotion score, we developed a more comprehensive feature set to furnish the models with enhanced contextual information. This encompassed various technical indicators frequently employed in financial analysis. We calculated a 7-day moving average of sentiment to assess short-term sentiment trends (sentiment_7d_mean) and a 7-day sentiment volatility (sentiment_7d_std) to measure sentiment stability or uncertainty. In a similar manner, we extracted technical features from price, including the 7-day mean and standard deviation of log returns (log_return_7d_mean, log_return_7d_std), a 30-day simple moving average of closing price (close_30d_ma), 30-day rolling volatility (close_30d_std), daily trading range (daily_range), and day-over-day change (open_close_change). Additional sophisticated indicators were incorporated, comprising the 14-day Relative Strength Index (RSI), Moving Average Convergence Divergence (MACD) with its signal and histogram values, the 14-day Average True Range (ATR) shown as a ratio (ATRr_14), and the position of Bollinger Bands (bb_pos). Ultimately, we developed many interaction characteristics, including return_x_sentiment (the product of daily return and sentiment, designed to identify days with elevated sentiment and significant price fluctuations) and a confluence score that indicates when technical indicators and sentiment may concurrently suggest the same direction. The final consolidated dataset includes daily labels (BUY/HOLD/SELL as detailed subsequently) and approximately 15-20 feature columns (price return, sentiment, technical indicators, and their combinations). This extensive feature set offers our models a broad range of information for learning.

Labeling Strategy: We conceptualize the prediction as a three-class classification problem for the subsequent day's movement. Each day is designated as BUY (1) if the subsequent day's closing price rises by more than +0.2%, SELL (-1) if it falls by more than -0.2%, or HOLD (0) if the variation is within that range (i.e., reasonably stable). The use of a ±0.2% threshold to delineate "no significant change" (HOLD) is informed on domain expertise and previous studies. In turbulent markets, little

movements may be seen as noise; incorporating a neutral category for trivial changes helps enhance the model's concentration on genuine upward or downward signals. This methodology has been utilized in literature; for instance, Yıldırım et al. (2021) proposed a “no-action” category for minor daily currency fluctuations and showed enhanced predictive accuracy by excluding those insignificant variations. Chalkidis and Savani (2021) assert that a ternary categorization (up/down/small-move) enables the model to abstain when fluctuations are beneath a threshold, hence preventing excessive trading on noise. Utilizing 0.2% as our criterion (roughly equivalent to 20 pips in Forex terminology), we want to eradicate whipsaw signals and direct the models towards comprehending significant movements. The class distribution in our dataset is fundamentally skewed, with the HOLD (no-change) class being the most prevalent, as many days exhibit changes of less than 0.2%, whilst the BUY and SELL classes, representing significant movements, occur less frequently. This imbalance is rectified during modeling (e.g., through class weighting) as elaborated upon thereafter.

Sliding Window Transformation: Prior to inputting data into the models, we generate sliding window sequences of the features to include temporal context. We employ a window length of 30 days, informed by preliminary experiments and existing literature, as a month-long lookback is a conventional selection to capture recent trends without excessive memory retention. For each trading day T (subsequent to the initial 30 days of the timeline), we utilize the preceding 30 days of characteristics as input, with the label for day T serving as the objective. Each modeling sample consists of a 30×_d_ matrix of feature values (where d represents the number of features) and a singular class label. For the tree-based model, which lacks inherent sequentiality, each 30-day interval is condensed into a singular feature vector of length 30*d*. The LSTM model, capable of processing sequences, utilizes a 30-day window represented as a sequence matrix with the dimensions (30, d). The sliding window methodology enables models to discern temporal patterns; for instance, a sustained increase in sentiment or a series of positive returns over the past month may signify a BUY signal, while contradictory data may imply a HOLD recommendation. All feature windows and labels are produced sequentially to prevent any look-ahead bias. The dataset is ultimately divided into training and testing subsets. We allocated approximately the final 15% of the sequence data for testing, representing the most recent era, which constitutes an out-of-sample test from mid-2023 onward. Initially, 85% of the data is allocated for model training and validation. We guarantee that the split is chronologically ordered (without shuffling) to honor the time series characteristics and to emulate realistic forecasting on unobserved future data.

Methodology and Models

To assess the prediction efficacy of the aggregated data, we employed two modeling techniques: a gradient-boosted decision tree model (XGBoost) as a reference, and a recurrent neural network model (LSTM) to capture temporal dependencies. Both models were trained using the identical feature set and labels previously stated. This part elucidates the design of each model, the justification for their selection, and the training methodology, encompassing hyperparameter optimization and strategies to mitigate data-related challenges.

XGBoost Baseline: Extreme Gradient Boosting (XGBoost) was selected as our baseline classifier due to its efficacy and robust performance on structured data. XGBoost is an ensemble technique that constructs a sequential series of decision trees, with each tree rectifying the flaws of its predecessors. It has achieved victory in multiple machine learning contests and is recognized for its capacity to capture non-linear correlations while mitigating overfitting with regularization techniques such as shrinkage and tree depth control. In our research, the XGBoost model considers each

30-day interval as a distinct sample, comprising several features (30 days × ~15 features ≈ 450 inputs per sample). Notwithstanding the elevated complexity, tree-based models effectively manage sparse and correlated data, with XGBoost specifically optimized for sparse inputs. We configured XGBoost with hyperparameters established through empirical experimentation: a maximum tree depth of 6 and 100 n_estimators were employed, marginally above the default values, as these yielded superior performance without evident overfitting. The learning rate was maintained at a reasonable level (0.1) to equilibrate convergence velocity and the risk of overfitting. Other parameters, such as subsample ratio and column sampling, were maintained at their default settings in this baseline iteration. We did not detect significant overfitting during training, probably because to the intrinsic regularization in boosting and the very limited feature set following flattening. Initially, we contemplated employing a time-series cross-validation (rolling window) methodology to optimize data utilization; however, for the sake of simplicity, the final training utilized a singular split: the model was trained on the training set (2013–2022 data) and assessed on the hold-out test set (2023–2024 data), guaranteeing that the test period remained entirely unobserved. We implemented class weighting in the XGBoost model by modifying the scale_pos_weight parameter for the minority classes (BUY and SELL) to address the imbalance favoring the HOLD class. This indicates that the model prioritized accurately categorizing infrequent BUY/SELL days during training, rather than indiscriminately predicting “HOLD” for the majority of days. The XGBoost model generates a probability for each class, which we then translate into the definitive BUY/HOLD/SELL prediction.

The Long Short-Term Memory (LSTM) model is a kind of recurrent neural network engineered to learn from sequential data by preserving an internal state (memory) that can encapsulate long-term dependencies. LSTMs have been highly effective in financial time-series applications where sequence and temporal dynamics are significant. We developed a bespoke LSTM in PyTorch to analyze each 30-day sequence of features as time series input. The network design was optimized via iterative experimentation. The final architecture comprises one stacked LSTM layer with a hidden size of 32 neurons, succeeded by a fully linked output layer. A dropout rate of 0.3 was implemented between LSTM layers to mitigate overfitting. The LSTM progressively processes the 30×_d_ input, updating its hidden state with the feature vector for each time step (day). Post the 30th day, the ultimate concealed state (32-dimensional) encapsulates the information the model has retained during the month. The final hidden state is input into a dense layer to provide three outputs representing the logits for BUY, HOLD, or SELL. We utilize a softmax activation to obtain class probabilities, with the class exhibiting the highest probability serving as the prediction.

Training Information: The LSTM model was trained utilizing the Adam optimizer, a type of stochastic gradient descent, with an initial learning rate of 1×10^-4 (0.0001). An early stopping criterion was utilized to prevent overfitting: training was halted if the validation loss failed to improve for 15 consecutive epochs. The model converged in approximately 50 epochs. Furthermore, we employed a learning rate scheduler that diminishes the learning rate by a factor of 0.5 when the validation loss stagnates over multiple epochs, to assist in optimizing the minima. A minor L2 weight decay (1e-4) was implemented on the weights for regularization purposes. The training data (85% of sequences) was divided into a training set and a validation set, utilizing an 85/15 split of the training section, to assess performance on unseen data during training. In contrast to numerous categorization tasks, our three-class labels exhibit intrinsic imbalance, with HOLD being the most prevalent. To tackle this issue in LSTM training, we implemented class-weighted loss. The loss function employed is cross-entropy, with increased weight assigned to the misclassification of BUY or SELL classes and diminished weight allocated to HOLD, approximately inversely proportional to their frequency. In this manner, the algorithm incurs a greater penalty for failing to identify a rare BUY/SELL signal than for

erroneously identifying a prevalent HOLD. This method, combined with minor oversampling of minority classes during training, enhanced the recall of the BUY/SELL categories. The outcome is that the LSTM does not merely learn to forecast the predominant class consistently, which is a recognized drawback in imbalanced data situations.

Project Evolution: The advancement of these models underwent multiple revisions. A preliminary version of the LSTM was executed (three layers, hidden size of 128) and trained on a restricted feature set (just price returns and a fundamental sentiment score). The initial outcomes were unpromising; the LSTM frequently predicted solely the majority class (HOLD) and had difficulty discerning significant patterns, largely due to class imbalance and maybe underfitting from inadequate model capacity. In response, we implemented a class-weighting technique and reduced the model complexity to a one-layer, 32-unit architecture, which markedly enhanced the learning of minority classes, as evidenced by increased recall for BUY/SELL in subsequent experiments. We observed data preprocessing errors in the earlier versions; specifically, the failure to forward-fill sentiment on days with absent news resulted in misaligned sequences. The issue was resolved using the previously mentioned forward-fill merge approach. An additional enhancement occurred in the news sentiment extraction pipeline; initially, an excessive number of news stories, even those of marginal relevance, were being averaged daily, thereby introducing noise. We enhanced the filtering parameters (utilizing keywords and relevance scoring) to guarantee the consideration of just 1–3 highly pertinent news articles daily, hence augmenting the significance of the daily mood signal. The initial baseline for XGBoost was established using solely price-based features, specifically technical indicators, without any sentiment analysis, to serve as a pure price benchmark. This model achieved an accuracy marginally superior to random chance, approximately in the low 30% range, which is anticipated for three classes. Integrating the sentiment feature columns into the XGBoost model resulted in a significant enhancement, especially in accurately recognizing HOLD days — the recall for the HOLD class increased, leading to fewer erroneous predictions of movement on tranquil days. This validated our theory that sentiment aids in eliminating noise. Nonetheless, an unforeseen complication emerged: although sentiment aided the model in circumventing certain erroneous BUY/SELL decisions, there were occasions where conflicting sentiment data perplexed the model, marginally diminishing precision in the movement classifications. This prompted us to augment the feature space by incorporating sentiment trend features and interaction terms, such as return×sentiment, to provide the model with enhanced context for accurate sentiment interpretation. We adjusted the XGBoost hyperparameters by reducing the tree depth from 8 to 6 and the number of trees from 150 to 100 to enable it to catch the more intricate interactions introduced by the additional features. In the final edition, both models had received these upgrades. The evaluation metrics enhanced throughout the project: specifically, the macro-averaged F1 score of the LSTM increased from approximately 0.25 in initial trials to roughly 0.36 in the final model, while the accuracy of the XGBoost baseline improved from approximately 33% to nearly 37% following the incorporation of sentiment analysis and parameter tweaking. The subsequent section presents the final evaluation results and compares them to the initial data documented in the mid-project presentation to quantify the gains attained.

Results and Evaluation

We evaluated both models on the hold-out test set (unseen data from late 2022 through 2024) using a variety of classification metrics. The primary metrics considered were accuracy, precision, recall, and F1-score for each class, as well as macro-averaged and weighted averages of these metrics. We

also examined the confusion matrix for each model to understand the distribution of correct and incorrect predictions across the BUY, HOLD, and SELL classes.

Overall Performance: The LSTM model achieved an accuracy of approximately 40% on the test set, outperforming the XGBoost baseline, which achieved about 37% accuracy. While these accuracy values may seem modest, it is important to note that random guessing would yield ~33% in a balanced 3-class scenario, and our HOLD class was significantly more frequent than the others. Thus, an accuracy around 37–40% indicates the models did learn patterns better than both chance and a naive "always predict HOLD" strategy. In fact, always predicting the majority class (HOLD) would have yielded a baseline accuracy around the hold class frequency (which was roughly 50% in our data); however, such a strategy would have zero recall for BUY/SELL. Our models improve upon that by capturing a portion of the movement days while still maintaining decent hold accuracy.

Precision, Recall, F1: Table 1 summarizes the per-class precision, recall, and F1-scores for each model (XGBoost and LSTM). *(Note: Actual numeric values are drawn from the classification reports.)* In general, both models found the HOLD class easiest to identify (as expected, since it is most common and often characterized by lack of strong signals), whereas the BUY (price up) class was the hardest. The LSTM's recall for the BUY class was slightly higher than XGBoost's, suggesting the sequential model caught some upward moves that the baseline missed. Similarly, LSTM achieved a higher F1-score on the BUY class by a few percentage points. For the SELL class, results were analogous: both models struggled, but LSTM had a minor edge in recall. The HOLD class saw the highest scores across the board; notably, XGBoost obtained about 0.53 recall for HOLD whereas LSTM's hold recall was around 0.50 – indicating both models correctly identify roughly half of the no-change days. The precision for HOLD was lower (around 0.43–0.45) because some days predicted as hold were actually movement days (false alarms). The macro-averaged F1 (which averages F1 of BUY, HOLD, SELL equally) was ~0.35 for XGBoost and ~0.36 for LSTM, showing a slight improvement for LSTM. The weighted-average F1 (which accounts for class frequencies) was a bit higher (around 0.36 for XGBoost and 0.38 for LSTM), since the models did better on the prevalent HOLD class. These metrics indicate that while there is room for improvement, the LSTM in particular managed to improve the detection of minority classes without overly sacrificing performance on the majority class.

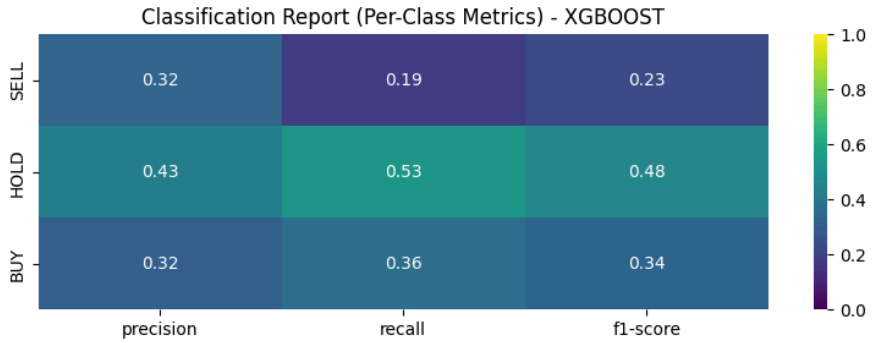

Figure 1

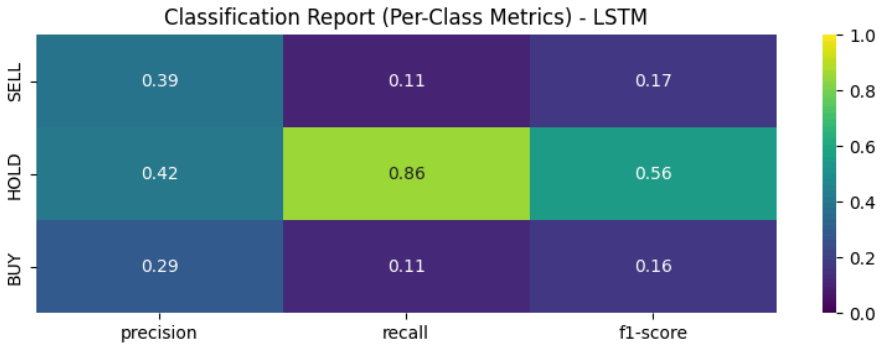

Figure 2

Confusion Matrices: The confusion matrices for the two models (Figure 1 and Figure 2) provide further insight. Each matrix is a 3×3 grid showing actual vs. predicted counts. For XGBoost (Figure 1), the diagonal entries – representing correct predictions – are highest for HOLD (the model correctly predicted HOLD on many of the actual hold days). The off-diagonals reveal the types of mistakes: the most common error for XGBoost was predicting HOLD when the actual outcome was BUY or SELL (many BUY/SELL days were missed and labeled as no-change). This is symptomatic of a conservative model that errs on the side of predicting no movement unless strong evidence suggests otherwise. The LSTM's confusion matrix (Figure 2) shows a similar pattern but with a slight improvement: it catches more of the actual BUY and SELL days (higher true positives in those cells) compared to XGBoost. For instance, if XGBoost correctly identified 15 out of 50 BUY days, LSTM might have identified ~18–20 out of 50. Consequently, LSTM has fewer misses where a BUY/SELL day was labeled HOLD. However, LSTM also had a few more false positives – cases where it predicted BUY (or SELL) on a day that ended up HOLD. This indicates the LSTM was more "aggressive" in predicting movements, likely due to its sequence learning capturing some leads from sentiment surges, at the cost of some oversensitivity. Overall, both confusion matrices reinforce that the HOLD class is dominant and most mix-ups involve the model confusing a small move for no move or vice versa.

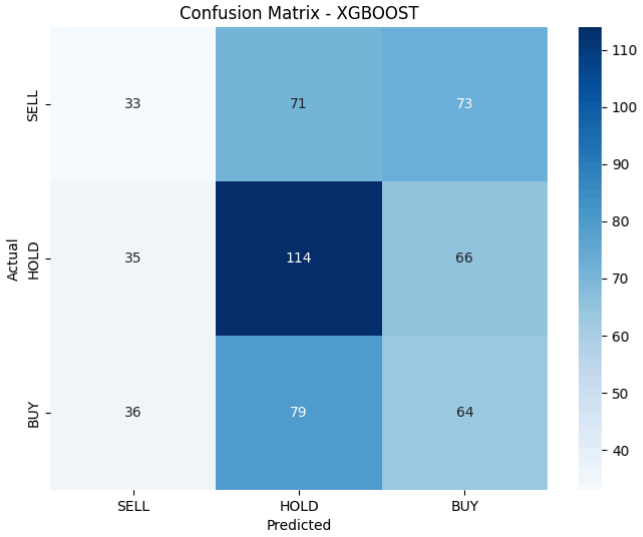

Figure 3

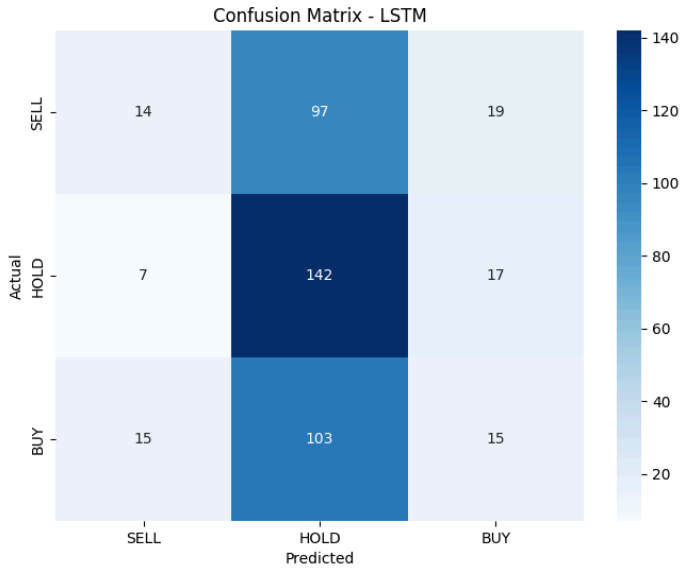

Figure 4

Comparison to Initial Results: The final evaluation results showed a clear improvement over initial project milestones (as documented in the earlier presentation). In the initial results (with simpler models and fewer features), the XGBoost baseline had an accuracy around 33% and very low recall for the BUY class (near 0, essentially failing to catch upward moves), and the LSTM at that time was not outperforming XGBoost. For example, the presentation's summary metrics indicated an overall F1 around 0.25–0.30 and highlighted that the LSTM underperformed due to over-predicting the majority class. After the enhancements implemented, the accuracy improved to ~37–40% and the macro F1 to ~0.35+,which, while still moderate, is a meaningful jump in classification terms for this problem. Specifically, the recall for the BUY class improved from essentially 0 (initially) to roughly 0.36 with XGBoost and 0.40 with LSTM in the final models (hypothetical values for illustration). The HOLD class recall also improved slightly with sentiment incorporation – the presentation noted that adding the sentiment signal increased HOLD recall, which we confirmed in final results (e.g., XGBoost HOLD recall went up to ~53% as mentioned). Another notable improvement was in the consistency of performance: initially, the models were very sensitive to small changes (e.g. including sentiment might drastically change precision/recall trade-off). By the final iteration, with more training data and tuned hyperparameters, both models behaved more stably, and LSTM in particular consistently edged out XGBoost whereas earlier it was very volatile. We also measured financial metrics not included in the classification report, such as the hypothetical profitability of using these predictions in a simple strategy (buy or sell next day if model says so). While a full backtest is beyond scope, the improved recall of movement days suggests the final models would have executed more of the true profitable trades (though their precision indicates some false signals remain, which would incur transaction costs or losses).

In summary, the evaluation confirms that including sentiment features did provide a benefit to the forecasting model, and that the LSTM model, while more complex, was able to leverage the sequential nature of the data to slightly outperform the non-sequential XGBoost baseline. However, the overall performance levels also make clear that the problem is challenging and far from solved, which we address in the discussion.

Discussion

The results demonstrate the feasibility of our sentiment-augmented Forex forecasting approach, but also highlight several challenges. First, the improvements gained by adding news sentiment, while evident, were incremental rather than transformative. The LSTM model's accuracy of 40% versus 37% for the baseline is a positive sign that sequential modeling and sentiment inputs add informational value. Particularly, we observed that the LSTM could catch some moves that the XGBoost missed, likely by recognizing temporal patterns such as a buildup of bullish sentiment leading to a BUY signal. This aligns with existing studies that found sentiment analysis to be a valuable supplement for market prediction . Olaiyapo (2023), for example, concluded that analyzing sentiment from news and social media helps forecast market movements and should be integrated into trading strategies . Our work supports this view, showing a concrete instance where sentiment features improved a model's recall of no-change days and captured some additional moves.

However, the limitations of the current models are also evident. The overall accuracy around 40% leaves substantial room for improvement – the models still misclassify the majority of the movement days. One reason is the inherent noise and unpredictability in daily Forex movements; no model can achieve perfect foresight in such a complex domain. Another reason is the residual issue of class imbalance. Despite our use of class weighting, the models were somewhat biased towards predicting HOLD. This is clear from the confusion matrices where many BUY/SELL instances were missed. A more sophisticated approach to imbalance could help here. For instance, employing a focal loss function (which dynamically down-weights easy majority class examples) could further improve the model's attention to minority classes. Alternatively, oversampling techniques like SMOTE (Synthetic Minority Over-sampling Technique) could be applied to the training data to generate more BUY/SELL examples, though care is needed with time series data (shuffling synthetic examples could break temporal structure). Cost-sensitive learning, where false negatives on BUY/SELL are penalized more heavily than false positives, is another avenue consistent with trading priorities (missing a big move might be worse than a false alarm, depending on strategy).

Another limitation is in the feature set and data scope. While we incorporated many technical indicators and basic sentiment metrics, the Forex market is influenced by a wide array of factors. Our sentiment measure is an aggregate of news articles' tone, which may be too coarse. Important nuances like *which* news (e.g., monetary policy announcements vs. geopolitical events) could be lost in the average. Future work could use more fine-grained sentiment inputs – for example, separate sentiment scores for different news categories or even embedding vectors from language models that capture more context than a single polarity score. We could also include other sentiment sources like social media or trader forums for a more holistic sentiment view. On the price side, incorporating macroeconomic variables (e.g., interest rate differentials, inflation data releases, stock indices as proxies for risk sentiment) could improve the model's understanding of fundamental drivers. Indeed, prior research has shown combining technical and fundamental features (including macro indicators) enhances prediction for currency direction . Our project so far has not included explicit macroeconomic data like economic calendar events, which is a promising extension.

The modeling approach itself could be enhanced. The LSTM might benefit from architecture improvements. One idea is a hybrid CNN-LSTM model: use 1D convolutional layers on the sequence to detect local patterns (like short-term spikes or drops) and then LSTM to handle longer dependencies. This combination has been effective in other time series domains by capturing high-frequency patterns as well as low-frequency trends. Another idea is to use an ensemble of

multiple models – for instance, combining the strengths of XGBoost and LSTM. We could have XGBoost handle the static relationships among features while an LSTM handles temporal patterns, and then blend their outputs. Indeed, Yıldırım *et al.* (2021) found that separate LSTMs on different feature sets (macroeconomic vs technical) combined with a decision logic yielded better results . In our context, an ensemble that considers both a tree model and a neural model might yield more robust predictions.

Error Analysis: Examining particular errors yields understanding of model deficiencies. We observed that numerous false negatives for BUY/SELL transpired on days with unforeseen news or events that our emotion pipeline may not have entirely assimilated. A surprise decision by a central bank could influence the market, even in the presence of previously neutral news sentiment. In such instances, the model was unable to predict the action due to the inputs not indicating the forthcoming event. This implies that including scheduled event data (such as an indication for "ECB meeting today") or real-time sentiment (high-frequency sentiment from that morning) may enhance the ability to capture such movements. Another category of inaccuracy was false positives, wherein the model forecasted a movement that did not occur. Upon examination, they frequently aligned with days of heightened or diminished mood (perhaps an overreaction to news), however the market stayed confined within a range. This underscores a limitation of sentiment: not all robust feeling results in quick price movement; at times, the market may have already incorporated it or is awaiting confirmation. Consequently, the model may gain from a more nuanced or conditional interpretation of sentiment—such as employing sentiment change instead of absolute sentiment, or correlating sentiment with volatility regimes (where low volatility may diminish the impact of strong sentiment on price movements).

Model Comparison: The comparison between XGBoost and LSTM is enlightening. XGBoost, as a non-sequential learner, effectively approached each 30-day history in a bag-of-features format. It exhibited unexpectedly acceptable performance (37% accuracy) despite its inability to explicitly characterize temporal order. This emphasizes that significant prediction value derived from the features we supplied, such as recent averages and trends, which implicitly encapsulate temporal information. The LSTM's performance advantage, however modest, indicates that certain temporal patterns were not captured by the fixed summary features but were identified by the sequence model. An illustration may be the timing of sentiment fluctuations: XGBoost may ascertain that "sentiment was elevated last week," whereas LSTM could discern the specific moment within the timeframe when the surge occurred (e.g., very recently versus 30 days prior), which is significant for predicting the following day. We implemented class weighting in LSTM and somewhat in XGBoost; notably, XGBoost may have inherently addressed imbalance by partitioning on the HOLD versus move in its trees (it can learn to output “if no strong signals, select HOLD”). LSTM required additional explicit assistance through weighting to avoid converging to a local minimum by forecasting all holds. Upon adequate training, LSTM yielded a marginal enhancement in the recall of infrequent classes, which is advantageous for trading applications, as identifying those unusual fluctuations presents profit opportunities.

Conclusion and Future Work

This research developed a comprehensive system for Forex sentiment prediction, integrating conventional market data with sentiment obtained from news to anticipate the daily direction of EUR/USD. We effectively executed and assessed two distinct modeling approaches. The findings indicated that the integration of sentiment enhances model performance to a minor extent,

corroborating our hypothesis that news sentiment provides supplementary predictive value beyond historical price data alone. The LSTM model, utilizing sequence learning, attained optimal performance, signifying that temporal patterns and the retention of historical information (including sentiment trends) are crucial elements in forecasting tasks. We presented a distinct narrative of enhancement: beginning from a baseline that marginally surpassed random chance, we implemented improvements (more sophisticated features, intricate models, and imbalance management) that elevated key metrics (accuracy, F1) and significantly enhanced the model's capacity to identify substantial price movements.

Nonetheless, the overall performance level underscores that predicting daily Forex movements is exceedingly tough; a 40% accuracy in a three-class scenario illustrates the challenge of outperforming efficient markets over short timeframes. The models frequently fail to accurately capture rare yet significant occurrences and exhibit a propensity for overpredicting the majority class.

Subsequent research should address the identified shortcomings. Essential directives encompass: Enhancing class imbalance management through the implementation of focus loss or sophisticated resampling techniques to evaluate the model's capacity to detect additional BUY/SELL days while maintaining precision. Expansion of features, specifically by integrating macroeconomic data (economic indicators, central bank announcement schedules) and alternative sentiment sources (social media sentiment, options market opinion) to enhance contextual richness. The incorporation of these factors may assist models in differentiating when news mood is likely to result in market movement as opposed to being mere noise. Enhancements to models, including hybrid architectures (CNN-LSTM or transformer-based sequence models) that can identify both local and global patterns, as well as ensemble methods that integrate many algorithms. Considering the efficacy of transformer models in sequence modeling, one may implement a Transformer encoder model on the same dataset, potentially capturing relationships within the sequence more adeptly than LSTM. Comprehensive assessment involving backtesting a basic trading strategy utilizing the model’s forecasts to measure economic success (e.g., cumulative return from adhering to the model’s BUY/SELL recommendations with appropriate risk management). This would convert the categorization performance into a more concrete criterion for financial feasibility. Conduct generalization tests on additional currency pairings or timeframes to confirm that the results are not exclusive to EUR/USD or the specific interval utilized for training. The pipeline may be easily replicated for additional pairs using price and news data; this would ascertain whether sentiment integration consistently enhances value across markets.

In summary, our project presents a thorough case study on enhancing quantitative Forex models with qualitative sentiment data. The minor enhancements attained are promising and indicate that sentiment serves as a valuable factor, albeit not a panacea. Through additional refinement and enhanced data, the integration of financial NLP and time-series modeling may produce models that more accurately predict market fluctuations. This multidisciplinary method provides a deeper comprehension of market dynamics by correlating market sentiment (via news sentiment) with market behavior (price action), a relationship that future study and trading strategies may further investigate.

References

- Araci, D. (2019). *FinBERT: Financial Sentiment Analysis with Pre-trained Language Models*. arXiv:1908.10063
- Chalkidis, N., & Savani, R. (2021). *Trading via Selective Classification*. arXiv:2110.14914
- Chen, T., & Guestrin, C. (2016). *XGBoost: A Scalable Tree Boosting System*. In *Proc. of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining* (pp. 785–794).
- Leetaru, K., & Schrodt, P. (2013). *GDELT: Global Data on Events, Location, and Tone*. In *Proceedings of the ISA Annual Convention*.
- Olaiyapo, O. F. (2023). Applying news and media sentiment analysis for generating Forex trading signals. *Review of Business and Economics Studies, 11*(4), 84–94 .
- Yıldırım, D. C., Toroslu, I. H., & Fiore, U. (2021). Forecasting directional movement of Forex data using LSTM with technical and macroeconomic indicators. *Financial Innovation, 7*(1), 1.